

# Snackbar

Snackbar A flexible, accessible snackbar component for transient notifications, actions, and feedback.

Supports multiple positions, variants, icons, actions, and theming. Use snackbars to provide brief messages about app processes, confirmations, or errors, with optional actions. How to use ■ **import { SnackbarService } from "@aava/play-core" ; Basic Usage ■ Show a simple snackbar with a message. Angular Preview Code**

```
import { Component , inject } from "@angular/core" ; import { CommonModule } from "@angular/common" ;
import { SnackbarService , AavaButtonComponent } from "@aava/play-core" ; @ Component ( { standalone :
true , imports : [ CommonModule , AavaButtonComponent ] , template : ` <div> <aava-button label="Show
Snackbar" variant="primary" size="md" state="default" (click)="showSnackbar()" > </aava-button> </div> ` , } )
export class SnackbarBasicUsageDemoComponent { private snackbar = inject ( SnackbarService ) ;
showSnackbar ( ) { this . snackbar . show ( "This is a snackbar message" , "top-center" ) ; } } Positions ■
```

Display snackbars in different screen positions. Angular Preview Code

```
import { Component , inject } from
"@angular/core" ; import { CommonModule } from "@angular/common" ; import { SnackbarService ,
SnackbarPosition } from "@aava/play-core" ; import { AavaButtonComponent } from "@aava/play-core" ; @
Component ( { standalone : true , imports : [ CommonModule , AavaButtonComponent ] , template : ` <div>
<aava-button *ngFor="let pos of positions" [label]="pos" variant="primary" size="md" state="default"
(click)="showSnackbar(pos)" > </aava-button> </div> ` , } ) export class SnackbarPositionsDemoComponent {
private snackbar = inject ( SnackbarService ) ; positions : SnackbarPosition [ ] = [ "top-left" , "top-center" ,
"top-right" , "bottom-left" , "bottom-center" , "bottom-right" , "center" , ] ; showSnackbar ( position :
SnackbarPosition ) { this . snackbar . show ( ` Snackbar at ${ position } ` , position ) ; } } There are 7 common
positions used to describe snackbars placement. top-left : Positioned at the upper-left corner. top-center :
Positioned at the top edge, horizontally centered. top-right : Positioned at the upper-right corner. bottom-left :
Positioned at the lower-left corner. bottom-center : Positioned at the bottom edge, horizontally centered.
bottom-right : Positioned at the lower-right corner. center : Positioned exactly in the middle, both vertically and
horizontally centered. Actions & Icons ■ Add action buttons, icons, make snackbars dismissible or persistent.
Angular Preview Code API Reference ■ Inputs ■ Property Type Default Description message string " Message
text to display in the snackbar position SnackbarPosition 'bottom-center' Position of the snackbar on screen
duration number 4000 Duration in milliseconds before auto-dismiss color string " Custom text color for the
snackbar backgroundColor string " Custom background color for the snackbar action SnackbarAction " Action
button configuration icon SnackbarIcon " Icon configuration for the snackbar dismissible boolean false Whether
the snackbar can be dismissed persistent boolean false Whether the snackbar persists until dismissed variant
'surface-bold' | string 'surface-bold' Visual variant of the snackbar type 'medium' | 'strong' | 'max' | string
'medium' Size/emphasis type of the snackbar Outputs ■ Event Type Description dismissed EventEmitter<void>
Emitted when snackbar is dismissed actioned EventEmitter<string> Emitted when action button is clicked
Methods ■ Method Parameters Return Type Description show() message: string, options?: SnackbarOptions
void Display a snackbar with the given message dismiss() void void Dismiss the currently displayed snackbar
clear() void void Clear all displayed snackbars CSS Custom Properties ■ Property Description
--snackbar-font-family Font family for snackbar text --snackbar-size-sm-font Font size for small snackbar
--snackbar-size-md-font Font size for medium snackbar --snackbar-size-lg-font Font size for large snackbar
--snackbar-success-color Color for success snackbar --snackbar-warning-color Color for warning snackbar
--snackbar-error-color Color for error snackbar --snackbar-info-color Color for info snackbar
--snackbar-font-weight-regular Regular font weight --snackbar-font-weight-semibold Semibold font weight
--snackbar-subtitle-color Color for subtitle text --snackbar-title-color Color for title text --snackbar-radius-sm
Border radius for snackbar --snackbar-shadowbox Box shadow for snackbar --snackbar-padding-small Padding
for small snackbar --snackbar-padding-medium Padding for medium snackbar --snackbar-padding-large
Padding for large snackbar --snackbar-padding-default Default padding for snackbar --snackbar-btn-font-sm
```

Font size for small button --snackbar-padding-small-btn Padding for small button --snackbar-btn-font-md Font size for medium button --snackbar-padding-md-btn Padding for medium button --snackbar-btn-font-large Font size for large button --snackbar-font-small Small font size --snackbar-font-medium Medium font size --snackbar-font-large Large font size --snackbar-font-large-title Font size for large title --snackbar-font-title Font size for title --snackbar-font-subtitle Font size for subtitle --snackbar-padding-wrapper Horizontal padding for snackbar container Accessibility ■ Keyboard accessible (focus, dismiss) Proper ARIA attributes for screen readers Focus management and visible indicators High contrast and reduced motion support Theming & Design Tokens ■ All colors, spacing, and effects are controlled via semantic design tokens and CSS custom properties. Override these in your theme or component styles for custom branding. Best Practices ■ Use snackbars for brief, non-blocking feedback Avoid placing critical information solely in snackbars Provide clear, actionable messages Use actions for undo or retry scenarios Test snackbar placement in responsive layouts Avoid excessive custom styling; use built-in variants and types

```
import { Component, inject } from "@angular/core";
import { CommonModule } from "@angular/common";
import { SnackbarService, AavaButtonComponent } from "@aava/play-core";

@Component({
  standalone: true,
  imports: [CommonModule, AavaButtonComponent],
  template: `
    <div>
      <aava-button
        label="Show Snackbar"
        variant="primary"
        size="md"
        state="default"
        (click)="showSnackbar()"
      >
      </aava-button>
    </div>
  `,
})
export class SnackbarBasicUsageDemoComponent {
  private snackbar = inject(SnackbarService);

  showSnackbar() {
    this.snackbar.show("This is a snackbar message", "top-center");
  }
}
```

```

import { Component, inject } from "@angular/core";
import { CommonModule } from "@angular/common";
import { MatSnackBarService, MatSnackBarPosition } from "@aava/play-core";
import { AavaButtonComponent } from "@aava/play-core";

@Component({
  standalone: true,
  imports: [CommonModule, AavaButtonComponent],
  template: `
    <div>
      <aava-button
        *ngFor="let pos of positions"
        [label]="pos"
        variant="primary"
        size="md"
        state="default"
        (click)="showSnackBar(pos)"
      >
      </aava-button>
    </div>
  `,
})
export class SnackBarPositionsDemoComponent {
  private snackbar = inject(MatSnackBarService);

  positions: MatSnackBarPosition[] = [
    "top-left",
    "top-center",
    "top-right",
    "bottom-left",
    "bottom-center",
    "bottom-right",
    "center",
  ];

  showSnackBar(position: MatSnackBarPosition) {
    this.snackbar.show(`SnackBar at ${position}`, position);
  }
}

```

■ No code found